

PATENT
450100-03139

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

APPLICATION FOR LETTERS PATENT

TITLE: DEMULTIPLEXER

INVENTORS: Hideki NABESAKO, Osamu YAGI

William S. Frommer
Registration No. 25,506
FROMMER LAWRENCE & HAUG LLP
745 Fifth Avenue
New York, New York 10151
Tel. (212) 588-0800

SPECIFICATION

Demultiplexer

BACKGROUND OF THE INVENTION

Field of the Invention

The present invention relates to a demultiplexer capable of separating desired packets from (a bit stream of) input digital data which has different types of packets multiplexed by a specific multiplexing method. More particularly, it relates to a demultiplexer provided with a command memory in which micro-codes are stored for controlling the action of each components in response to the micro-codes read out in a sequence from the command memory. As a result, the demultiplexer can handle different type of the multiplexing format through modifying the micro-codes read from the command memory, thus decreasing the overall size of its circuit and the overall cost.

Description of the Related Art

In general, video, audio, text, and other contents of digital data for the broadcasting and the storage mediums are encoded and multiplexed through grouping into packets or packs before transmitted and stored in the form of a stream of bits. When the packets in the received data are identical in the encoding format but

different the multiplexing format, the receiver has to prepare a plurality of demultiplexers for the different types of the multiplexing format.

For example, the technologies of DVB (digital video broadcasting), DSS (digital satellite system), and DVD (digital versatile disc) are different from each other in the multiplexing format.

Fig. 19 illustrates a common structure of the DVB packet in a DVB stream. The DVB packet comprises a header of 4 bytes, an adaptation field of a variable length, and a payload of a variable length (a data field) and its total length is 188 bytes. As well known, the payload contains PES (packetized elementary stream) packets separated again and assigned as well as various table of PSI (program specific information) in the form of sections determined by the MPEG2 system. The DVB packet may be composed of either the adaptation field or the payload. The header contains a PID (packet identification) data for the packet identification, while the adaptation field contains a PCR (program clock reference) data as the timing information.

Fig. 20 shows the contents of the header in the DVB packet. "Sync_byte" is 0x47. When "transport_error_indicator" is 1, it indicates that the packet has an error. When "payload_unit_start_indicator" is 1, it indicates that the payload in

the packet is marked with the PES or PSI header. When “transport_scrambling_control” is 00, it indicates that the packet is not scrambled. When the upper bit in “adaptation_field_control” is 1, it indicates that the packet contains the adaptation field. When the lower bit in the same is 1, it indicates that the packet contains the payload. “Continuity_counter” is provided for examining whether or not more than one of packets having the same PID data are arranged consecutively.

Fig. 21 illustrates the major contents of the adaptation field. “adaptation_field_length” indicates the length of the adaptation field. When “adaptation_field_control” is 10, 0xB7 (=183) is established. When “PCR_flag” is 1, it indicates that the adaptation field contains a PCR data as the timing information.

Fig. 22 illustrates a common structure of the DSS packet in a DSS bit stream. The DSS packet comprises a prefix of 2 bytes and a transport block of 128 bytes and its total length is 130 bytes. The prefix contains a SCID (service channel identification) data for the packet identification while the transport block contains the timing information.

Fig. 23 shows the major contents of the prefix in the DSS packet. “Packet Framing” is tuned to 0 or 1 alternately in each packet. When the packet is scrambled, “Control Flag” is 0. If not, 1. Fig. 24 illustrates one byte of a CC field and an HD field after the

prefix. "Continuity Counter" is provided for examining whether or not more than one of packets having the same SCID are arranged consecutively. "Header Designator" is provided for indicating the type of a video application packet.

For example, when the packet is an Auxiliary Data packet, "Header Designator" is 0000. Simultaneously, "Continuity Counter" is 0000. Fig. 25 shows 2 bytes after the CC field and the HD field in that case. When "Current Field Flag" is 1, it indicates that the Auxiliary Data packet is valid. "Aux Field ID" indicates what is contained in the Auxiliary Data packet. When 000000, the Auxiliary Data packet contains 5 bytes of a RTS (reference time stamp) data as the timing information. When 000011, the same packet contains the RTS data and an "Encryption Control Word Packet".

Fig. 26 illustrates a common structure of the DVD pack in a DVD bit stream. The DVD pack comprises a pack header of a variable length, a system header of a variable length, and a PES packet of a variable length and its overall length is variable. The PES header contains a stream ID as the packet ID and the pack header contains a SCR (system_clock_reference) data as the timing information.

Fig. 27 illustrates the major contents of the PES header. "PES_start_code_prefix" is always 0x000001, indicating the start of

the PES header. "PES_packet_length" indicates the length of the succeeding PES packet. "PES_header_data_length" indicates the length of "optical PES header" following "stream_ID".

Fig. 28 illustrates the major contents of the DVD pack header. "pack_start_code" is 0x000001BA. "system_clock_Reference_base" and "system_clock_Reference_extension" are the timing information. The system header follows the pack header of the first pack. Fig. 29 shows the major contents of the system header.

As the multiplexing formats are different, the construction of the streams, the contents of the headers, and the other settings become non-uniform. Accordingly, the header analysis and the payload transmission have to be modified depending on the multiplexing format.

Fig. 30 illustrates an arrangement of a demultiplexer 200 designed for separating desired packets for output from a bit stream of input digital data which has different types of packets multiplexed.

The demultiplexer 200 comprises an input terminal 201 for receiving a bit stream STM, a header analyzer 202 for analyzing the header of each packet or pack contained in the bit stream STM using a sequencer, an output destination determining unit 203 for determining the destination of outputting each packet, and a system

clock controller 204 for controlling a system clock with the timing information extracted from the bit stream STM in the header analyzer 202.

The output destination determining unit 203 has a built-in memory (not shown) thereof provided in which packet ID data have been registered for identifying the packets to be extracted from the bit stream STM. The packet ID may be received via a host interface 205 from an external CPU. When it is found by the header analyzer 202 that the packet ID extracted from a packet matches the packet ID stored in the built-in memory, the output destination determining unit 203 determines a destination identified by the stored packet ID as the destination of the packet having the extracted packet ID.

The demultiplexer 200 also includes a separator 206 for separating the destined packets from the bit stream STM and transferring them to the destination. The separator 206 may be connected to a set of output terminals 207a, 207b, 207c, and so on as the destinations.

When the bit stream STM is of the DVB format, the demultiplexer 200 shown in Fig. 30 carries out a procedure of packet processing shown in the flowchart of Fig. 31.

The procedure starts with receiving a DVB packet at Step ST11 and detecting a synchronous byte "sync_byte" in the packet at Step ST12. At Step ST13, the header is examined at

“transport_error_indicator”, “transport_scrambling_control”,
“adaption_field_control”, and the like whether or not they contain
any error. If not, the procedure goes to Step ST14.

At Step ST14, it is examined whether the PID data in the header
is identical to the PID data stored in the built-in memory of the
output destination determining unit 203. When the PID data in the
header is identical to the one stored, a continuous index
“continuity_counter” in the header is examined at Step ST15
whether or not the continuity of the packets is established. When
so, the procedure advances to Step ST16.

At Step ST16, it is examined whether the PCR data as the
timing information is contained or not. When the timing
information is contained, the procedure goes to Step ST17. At Step
ST17, the timing information is extracted from the packet and
transferred to the system clock controller 204. This is followed by
Step ST18. If the timing information is not found, the procedure
jumps to Step ST18. At Step ST18, the payload in the packet is
transmitted to the destination determined by the PID data of the
header before repeating the procedure for the succeeding DVB
packet.

When it is found at Step ST13 that the header contains an error,
the PID data of the header is not identified at Step ST14, and the
continuity is not found at Step ST15, the packet is discarded at Step

ST19 before repeating the procedure for the next DVB packet.

When the bit stream STM is of the DSS format, the demultiplexer 200 shown in Fig. 30 carries out a procedure shown in the flowchart of Fig. 32.

The procedure starts with receiving a DSS packet at Step ST21 and detecting a synchronous signal in the packet at Step ST22. At Step ST23, the prefix is examined for error bits (in "control_flag" or the like). If no error is found, the procedure goes to Step ST24.

At Step ST24, it is examined whether the SCID data in the prefix is identical to the SCID data stored in the built-in memory of the output destination determining unit 203. When the SCID data in the prefix is identical, "continuity_counter" in the prefix is examined at Step ST25 whether or not the continuity of the packets is established. When so, the procedure advances to Step ST26.

At Step ST26, it is examined whether the RTS (reference time stamp) data as the timing information is contained or not. When the timing information is contained, the procedure goes to Step ST27. At Step ST27, the timing information is extracted from the packet and transferred to the system clock controller 204. This is followed by Step ST28. If the timing information is not found, the procedure jumps to Step ST28. At Step ST28, the transport block in the packet is transmitted to the destination determined by the SCID data of the prefix before repeating the procedure for the succeeding

DSS packet.

When it is found at Step ST23 that the prefix contains an error, the SCID data is not identified at Step ST24, and the continuity is not found at Step ST25, the packet is discarded at Step ST29 before repeating the procedure for the next packet.

When the bit stream STM is of the DVD format, the demultiplexer 200 shown in Fig. 30 carries out a procedure shown in the flowchart of Fig. 33.

The procedure starts with receiving a DVD pack at Step ST31 and detecting a start code "pack_start_code" in the packet at Step ST32. At Step ST33, the SCR data is extracted as the timing information from the pack header and transmitted to the system clock controller 204. Then, the procedure goes to Step ST34.

At Step ST34, it is examined whether the pack is the first pack or not. When so, the system header is transmitted to the corresponding destination at Step ST35 and then, the procedure advances to Step ST36. If not the first pack, the procedure jumps to Step ST36.

At Step ST36, each PES packet is extracted and transmitted to the destination determined by the Stream ID in the header before repeating the procedure for the next pack.

As described above, for handling the signals of the different multiplexing formats which are different in the construction of a bit

stream and the analyzing process of each header, a corresponding number of the demultiplexers 200 are needed. Accordingly, the hardware circuitry arrangement for handling the different type of the multiplexing format with their dedicated circuits will be increased in the overall size and the production cost.

SUMMARY OF THE INVENTION

It is hence an object of the present invention to provide a demultiplexer capable of handling different types of the multiplexing format and decreasing the size and the cost of its overall circuit arrangement.

A demultiplexer for separating desired packets for output from an input digital data which has different format packets multiplexed in a given manner is provided comprising: a data input for receiving the input digital data; a first storage (a shift register) for storing and transferring the input digital data received at the data input; a second storage (a group of shift registers) for extracting and storing the headers of the packets from the input digital data stored in the first storage (a shift register); a calculating unit for analyzing the headers of the packets stored in the second storage (a group of shift registers); an output destination determining unit for determining the destination of the packets from a packet identifier which is contained in the headers of the packets stored in the second storage

(a group of shift registers); a separator arranged responsive to a result of the calculating action of the calculating unit and an output of the output destination determining unit for separating the desired packets from the input digital data received from the first storage (a shift register); a command memory for storing micro-codes provided for selecting a controlling action in each multiplexing format; a counter for determining an execution address of the micro-code stored in the command memory; a controller for controlling the action of each component with the micro-code read out from the command memory by the execution address determined by the counter; and a system clock counter for extracting the timing data from the input digital data stored in the first storage (a shift register) and controlling a system clock with the timing data.

In this embodiment, the command memory is provided for storage of the micro-codes. The micro-codes are read out in a sequence from the command memory and used for controlling the action of each component to separate desired packets from the input digital data and transfer them to the corresponding destinations. Accordingly, the different types of the multiplexing format can be handled by having a corresponding one of the micro-codes read out from the command memory, hence contributing to the reduction and the cost down of the circuitry arrangement of the demultiplexer.

BRIEF DESCRIPTION OF THE DRAWINGS

Fig. 1 is a block diagram showing an arrangement of a demultiplexer of the first embodiment;

Fig. 2 is a block diagram showing an arrangement of a demultiplexer of the second embodiment;

Fig. 3 is a diagram showing a structure of a content addressable memory;

Fig. 4 is a block diagram showing an arrangement of a demultiplexer of the third embodiment;

Fig. 5 is a block diagram showing an arrangement of a demultiplexer of the fourth embodiment;

Fig. 6 is a block diagram showing an arrangement of a demultiplexer of the fifth embodiment;

Fig. 7 is a block diagram showing an arrangement of a demultiplexer of the sixth embodiment;

Fig. 8 is a block diagram showing an arrangement of a demultiplexer of the seventh embodiment;

Fig. 9 is a diagram showing a processing action of the bit manipulating unit;

Fig. 10 is a block diagram showing an arrangement of a demultiplexer of the eighth embodiment;

Fig. 11 is a block diagram showing an arrangement of a demultiplexer of the ninth embodiment;

Fig. 12 is a diagram showing a shifting action of the shift register;

Fig. 13 is a block diagram showing an arrangement of a demultiplexer of the tenth embodiment;

Fig. 14 is a diagram showing a numbering action of the bytes;

Fig. 15 is a block diagram showing an arrangement of a demultiplexer of the eleventh embodiment;

Fig. 16 is a block diagram showing an arrangement of a demultiplexer of the twelfth embodiment;

Fig. 17 is a block diagram showing an arrangement of a demultiplexer of the thirteenth embodiment;

Fig. 18 is a block diagram showing an arrangement of a demultiplexer of the fourteenth embodiment;

Fig. 19 is a diagram showing a structure of the DVB packet;

Fig. 20 is a diagram showing major contents of the header in the DVB packet;

Fig. 21 is a diagram showing major contents of the adaptation field;

Fig. 22 is a diagram showing a structure of the DSS packet;

Fig. 23 is a diagram showing major contents of the prefix in the DSS packet;

Fig. 24 is a diagram showing a structure of a one-byte CC/HD field following the prefix in the DSS packet;

Fig. 25 is a block diagram showing a two-byte structure of an auxiliary data packet following the CC/HD field;

Fig. 26 is a diagram showing a structure of the DVD pack;

Fig. 27 is a diagram showing major contents of the PES header;

Fig. 28 is a diagram showing major contents of the DVD pack header;

Fig. 29 is a diagram showing major contents of the system header;

Fig. 30 is a block diagram showing an arrangement of a conventional demultiplexer;

Fig. 31 is a flowchart showing a procedure of processing the DVB packets;

Fig. 32 is a flowchart showing a procedure of processing the DSS packets; and

Fig. 33 is a flowchart showing a procedure of processing the DVD packs.

DESCRIPTION OF THE PREFERRED EMBODIMENT

Some embodiments of the present invention will be described referring to the relevant drawings.

For ease of the description, unless otherwise specified, the header of a DVB packet, the prefix of a DSS packet, and the header of a DVD pack are referred to as headers hereinafter. Also, the

packet identifier PID of a DVB packet, the packet identifier SCID of a DSS packet, and the packet identifier Stream ID of a PES packet in a DVD pack are referred to as packets ID. The timing data PCR of a DVB packet, the timing data RTS of a DSS packet, and the timing data SCR of a DVD pack are referred to as timing data. The payload of DVB packet, the transport block of a DSS packet, and the PES packet of a DVD pack are referred to as payloads.

Fig. 1 illustrates an arrangement of a demultiplexer 100A according to the first embodiment of the present invention. The demultiplexer 100A is arranged for separating desired packets from a bit stream of input digital data which have different types of packets multiplexed for output. The demultiplexer 100A can handle a variety of multiplexed data including the DVB stream, the DSS stream, and the DVD stream.

The demultiplexer 100A comprises an input terminal 101 for receiving a bit stream STM, a shift register 102 for saving and then transferring the bit streams STM received from the input terminal 101, a group of registers 103 for extracting and saving the headers from the bit stream saved in the shift register 102, a calculating unit 104 for analyzing the headers saved in the registers 103, and an output destination determining unit 105 for determining the destination of a packet marked with a packet ID which is carried in each header saved in the registers 103.

The demultiplexer 100A also includes a separator 106 arranged responsive to a result of the calculating unit 104 and a result of the output destination determining unit 105 for separating desired payloads from the bit stream STM received from the shift register 102 and delivering them to the destination. For example, the separator 106 is connected to a set of output terminals 107a, 107b, 107c, and so on. The shift register 102, the registers 103, the calculating unit 104, the output destination determining unit 105, and the separator 106 are connected with a bus 108.

The demultiplexer 100A further includes a command memory 111 for saving micro-codes for assigning particular controlling methods to the different types of the multiplexing, a counter 112 for obtaining the execution address of each micro-code saved in the command memory 111, a controller 113 for controlling the action of each component in response to the micro-code which is read out in a sequence from the command memory 111 with the use of data of the execution address from the counter 112, and a system clock controller 114 for extracting a timing data from the bit stream STM saved in the shift register 102 and using it to control the action of a system clock.

The action of the demultiplexer 100A shown in Fig. 1 will now be explained. In the demultiplexer 100A, the micro-codes corresponding to the multiplexing format of the bit stream STM are

read out in a sequence from the command memory 111 and used by the controller 113 controlling the action of each component, allowing the separator 106 to separate desired packets from the bit stream STM and transfer them to the predetermined destination.

The description is continued in case of the bit stream STM of a DVB format.

A bit stream STM received at the input terminal 101 is saved and shifted in the shifter register 102 by the action of a control signal which is produced by the controller 113 from the micro-code received from the command memory 111. The headers of packets in the bit stream STM saved in the shift register 102 are transferred to the group of the registers 103 when demanded. The headers of packets saved in the registers 103 are analyzed by the calculating unit 104 responding to a control signal from the controller 113. The analyzing process includes the detection of errors and the examination of continuity in the packets.

Upon receiving a control signal from the controller 113, the output destination determining unit 105 compares the PID data in the headers of packets saved in the registers 103 with reference PID data saved in its built-in memory to determine the destination of the packets. Using the result of the error detection and the continuity examination in the calculating unit 104 and an output of the output destination determining unit 105, the separator 106 separates and

transfers the payloads of the relevant packets to the destination.

The PCR data of timing information carried in the bit stream STM saved in the shift register 102 is transferred to the system clock controller 114 for controlling the system clock.

The demultiplexer 100A of this embodiment can perform the same process as of a flowchart of Fig. 31 when the bit stream STM is of the DVB format.

The action will be explained when the bit stream STM is of the DSS format.

A bit stream STM received at the input terminal 101 is saved and shifted in the shifter register 102 by the action of a control signal which is produced by the controller 113 from the micro-code received from the command memory 111. The prefixes (equivalent to the headers of the DVB packets) of packets in the bit stream STM saved in the shift register 102 are transferred to the group of the registers 103 when desired. The prefixes of packets saved in the registers 103 are analyzed by the calculating unit 104 responding to a control signal from the controller 113. The analyzing process includes the detection of errors and the examination of continuity in the packets.

Upon receiving a control signal from the controller 113, the output destination determining unit 105 compares the SCID data in the prefixes of packets saved in the registers 103 with reference

SCID data saved in its built-in memory to determine the destination of the packets. Using the result of the error detection and the continuity examination in the calculating unit 104 and an output of the output destination determining unit 105, the separator 106 separates and transfers transport blocks of the relevant packets to the destination.

The RTS data of timing information carried in the bit stream STM saved in the shift register 102 is transferred to the system clock controller 114 for controlling the system clock.

The demultiplexer 100A of this embodiment can perform the same process as of a flowchart of Fig. 32 when the bit stream STM is of the DSS format.

The action will be explained when the bit stream STM is of the DVD format.

A bit stream STM received at the input terminal 101 is saved and shifted in the shifter register 102 by the action of a control signal which is produced by the controller 113 from the micro-code received from the command memory 111. The headers of packs and the headers of PES packets (equivalent to the headers of DVB packets) in the bit stream STM saved in the shift register 102 are transferred to the group of the registers 103 when demanded. The headers saved in the registers 103 are analyzed by the calculating unit 104 responding to a control signal from the controller 113. The

analyzing process includes the detection of errors and the like.

Upon receiving a control signal from the controller 113, the output destination determining unit 105 compares the Stream ID data in the PES packets saved in the registers 103 with reference Stream ID data saved in its built-in memory to determine the destination of the packets. Using the result of the error detection in the calculating unit 104 and an output of the output destination determining unit 105, the separator 106 separates and transfers the PES packets to the destination.

The SCR data of timing information carried in the pack header of the bit stream STM saved in the shift register 102 is transferred to the system clock controller 114 for controlling the system clock.

The demultiplexer 100A of this embodiment can perform the same process as of a flowchart of Fig. 33 when the bit stream STM is of the DVD format.

The demultiplexer 100A shown in Fig. 1 is provided with the command memory 111 for saving the micro-codes. As the action of each component is controlled by the micro-codes read out in a sequence from the command memory 111, desired packets can readily be separated from the bit stream STM received at the input terminal 101 and delivered to the corresponding destination. This allows the different types of the multiplexing signal to be handled by modifying the micro-codes read out from the command memory 111,

hence contributing to the reduction of the overall size and the cost down of the circuit.

Fig. 2 illustrates an arrangement of a demultiplexer 100B according the second embodiment of the present invention. The demultiplexer 100B has an output destination determining unit 105 implemented by a contents addressable memory (CAM). The other components are identical to those of the demultiplexer 100A shown in Fig. 1.

Fig. 3 shows an arrangement of the contents addressable memory provided as the output destination determining unit 105. In this embodiment, the contents addressable memory can be retrieved with the packet ID of each header for reading out the destination and the preceding continuity count to detect the continuity.

Fig. 4 illustrates an arrangement of a demultiplexer 100C according to the third embodiment of the present invention. The demultiplexer 100C has an output destination determining unit 105 arranged of which the built-in memory can be written by an external CPU via a host interface 115. The other components are identical to those of the demultiplexer 100A shown in Fig. 1.

In this embodiment, the detailed data of packets in the bit stream STM is transferred to the external CPU. Upon analyzing the detailed data of packets, the external CPU can rewrite the data saved

in the built-in memory of the output destination determining unit 105.

Fig. 5 illustrates an arrangement of a demultiplexer 100D according to the fourth embodiment of the present invention. The demultiplexer 100D has an extra register 116 which serves as a counter for controlling the length of each packet. The other components are identical to those of the demultiplexer 100A shown in Fig. 1.

The DVB and DSS formats contain fixedly 188 bytes and 130 bytes respectively in each packet. When the length of the packet less the length of the header is registered to the register 116, its differential length of the payload or transport block can be output. As the length of the PES packet in the DVD packet is predetermined and registered in the register 116, it can be output.

Fig. 6 illustrates an arrangement of a demultiplexer 100E according to the fifth embodiment of the present invention. The demultiplexer 100E has a calculating unit 104 implemented by an arithmetic and logical unit (ALU). The other components are identical to those of the demultiplexer 100A shown in Fig. 1. For detection of any error and examination of the continuity, commands for addition, subtraction, logic sum, and logic product are released. The detection of any error and the examination of the continuity can be conducted by the arithmetic and logical unit.

Fig. 7 illustrates an arrangement of a demultiplexer 100F according to the sixth embodiment of the present invention. The demultiplexer 100F has a continuity examining unit 117 for examining the continuity between the packets. The other components are identical to those of the demultiplexer 100E shown in Fig. 6. The continuity examining unit 117 can simultaneously judge the continuity, the non-continuity, and the equality.

Fig. 8 illustrates an arrangement of a demultiplexer 100G according to the seventh embodiment of the present invention. This demultiplexer 100G has a bit handling unit 118 for handling bits. The other components are identical to those of the demultiplexer 100E shown in Fig. 6.

It is general in the analysis of the header that a particular part of the header is picked up and examined for error checking. The bit handling unit 118 carries out an action of picking the particular part of this data. As shown in Fig. 9, when the upper six to four bits are picked up from the data of eight bits stored in the register A (the least bit being 0), a masking of 01110000 can be used. For alignment of bits, the action of shift can be used. The bit handling unit 118 can perform those actions at one single cycle.

Fig. 10 illustrates an arrangement of a demultiplexer 100H according to the eighth embodiment of the present invention. The demultiplexer 100H has a calculating unit 104 implemented by an

arithmetic and logical unit (ALU), a continuity examining unit 117 for examining the continuity between packets, and a bit handling unit 118 for handling bits. The other components are identical to those of the demultiplexer 100A shown in Fig. 1.

Fig. 11 illustrates an arrangement of a demultiplexer 100J according to the ninth embodiment of the present invention. The demultiplexer 100J has an input buffer 119 connected between the input terminal 101 and the shift register 102 for temporarily saving the bit stream STM received at the input terminal 101. The other components are identical to those of the demodulator 100A shown in Fig. 1.

In the input buffer 119, the read action is carried out with the timing of an internal clock signal and the write action is carried out with the timing of an external clock signal synchronized with the transfer rate of the bit stream STM received at the input terminal 101. This allows the internal clock signal and the external clock signal to be timed with each other.

For example, the DSS packets and the DVB packets for the digital satellite broadcasting are substantially intercepted in the air. If the shift register 102 conducts a shifting action before a new data is received, unwanted data denoted (by the hatching) in Fig. 12 may be transmitted. For compensation, the shifting action of the shift register 102 is executed in synchronism with the reading of data

from the input buffer 119. This will inhibit the shift register 102 from releasing unwanted data.

Fig. 13 illustrates an arrangement of a demultiplexer 100K according to the tenth embodiment of the present invention. The demultiplexer 100K has a separator 106A thereof having an endian modifying function which replaces the separator 106 of the demultiplexer 100A of Fig. 1. The other components are identical to those of the demultiplexer 100A shown in Fig. 1.

The numbering of bytes in a word is classified into two major methods, as shown in Fig. 14, Big Endian mode for incrementing the number of each byte from the left to the right and Little Endian mode for incrementing the same from right to the left. The endian mode used in a decoder has to be changed from one to the other when the bit stream of video or audio data before the decoding process. The separator 106A can modify the endian mode depending on the number in the decoder.

Fig. 15 illustrates an arrangement of a demultiplexer 100L according to the eleventh embodiment of the present invention. In the other embodiments mentioned above, the payloads separated by the separator 106 are transmitted to the corresponding output terminals 107a, 107b, 107c, and so on determined by the output destination determining unit 105. Each stream of video or audio bits received at the output terminal may be transferred to a

dedicated decoder where they are decoded.

The demultiplexer 100L of this embodiment shown in Fig. 15 also includes an array of FIFO memories 120a, 120b, 120c, and so on which serve as the output buffer memories. Desired payloads are separated by a separator 106B from the bit stream STM received from the shift register 102 and saved in the corresponding FIFO memories assigned to the destinations determined by the output destination determining unit 105. The data of the payloads saved in the FIFO memories are then received by one single media processor 122 where they are decoded.

For action, the separator 106B transmits a relevant port address indicating the destination to an address decoder 121 which in turn generates and releases an enable signal to the FIFO memory designated. This allows the FIFO memories assigned to the destinations determined by the output destination determining unit 105 to save the payloads separated by the separator 106B.

Fig. 16 illustrates an arrangement of a demultiplexer 100M according to the twelfth embodiment of the present invention. While the demultiplexer 100L shown in Fig. 15 includes an array of the output buffer memories, the demultiplexer 100M of this embodiment has a single output buffer memory 123 implemented by a semiconductor memory, hard disk drive, or the like. The output buffer memory 123 includes an array of memory regions 123a, 123b,

123c, and so on. Desired payloads are separated by the action of a separator 106C from the bit stream STM received from the shifter register 102 and then stored in the corresponding memory regions assigned to the destinations determined by the output destination determining unit 105. The video or audio data stored in the output buffer memory 123 are then decoded in a single media processor 122.

In this embodiment, the output buffer memory 123 receives not only the payloads but also data of memory address from the separator 106C. The data of memory address allows the payloads received from the separator 106C to be stored in the corresponding memory regions assigned to the destinations determined by the output destination determining unit 105.

Fig. 17 illustrates an arrangement of a demultiplexer 100N according to the thirteenth embodiment of the present invention. The demultiplexer 100N has a command memory 111 arranged for receiving the micro-codes via a host interface 115 from an external CPU. The other components are identical to those of the demultiplexer 100A shown in Fig. 1.

The command memory 111 may save all the micro-codes assigned to the different types of the multiplexing format. However, while its size and production cost are increased, the command memory 111 will allow no rewrite function, thus being unfavorable

in the versatility. The demultiplexer 100N of this embodiment permits the command memory 111 to be downloaded with the corresponding micro-codes from the external CPU whenever the multiplexing format is changed from one to another. For example, when the multiplexing format is changed from the DVD format to the DVB format by the user, the micro-codes for the DVB format are downloaded from the external CPU into the command memory 111 via the host interface 115.

Fig. 18 illustrates an arrangement of a demultiplexer 100P according to the fourteenth embodiment of the present invention.

The demultiplexer 100P has an output destination determining unit 105 implemented by a contents addressable memory (CAM) similar to that of the demultiplexer 100B shown in Fig. 2. Accordingly, as the contents addressable memory is accessed by the packet ID in the header, its contained data of the preceding continuity count can be picked up for examination of the continuity with the destination.

The demultiplexer 100P like the demultiplexer 100C shown in Fig. 4 allows the built-in memory of the output destination determining unit 105 to be rewritten via the host interface 115 from the external CPU. Accordingly, as the external CPU receives and analyzes the detailed information of the packets in the bit stream which configure the bit stream STM, it can rewrite the contents in

the built-in memory of the output destination determining unit 105 at this external CPU.

The demultiplexer 100P includes a register 116 which acts as a counter for managing the length of each packet, similar to that of the demultiplexer 100D shown in Fig. 5, a continuity detector 117 for detecting the continuity between packets, similar to that of the demultiplexer 100F shown in Fig. 7, and a calculating unit 104 implemented by an arithmetic logic unit (ALU), similar to that of the demultiplexer 100E shown in Fig. 6.

The demultiplexer 100P like the demultiplexer 100G shown in Fig. 8 also includes a bit manipulating unit 118 for manipulating bits. This allows any desired portion in the header to be extracted with ease before examined in the calculating unit 104 for error checking.

The demultiplexer 100P like the demultiplexer 100J shown in Fig. 11 also includes an input buffer 119 connected between the input terminal 101 and the shift register 102 for temporarily saving the bits stream STM received at the input terminal 101. In the input buffer 119, the read action is carried out with the timing of an internal clock signal and the write action is carried out with the timing of an external clock signal synchronized with the transfer rate of the bit stream STM received at the input terminal 101. At the timing of the input buffer 119 receiving the bit stream STM, the

reading of data from the input buffer 119 is performed and the shift register 102 is shifted, hence inhibiting unwanted data from being transmitted across the shift register 102.

The demultiplexer 100P like the demultiplexer 100K shown in Fig. 13 also includes a separator 106D which has an endian modifying function. The separator 106D can modify the endian according to the method of numbering in a decoder for decoding the bit stream of video or audio data.

The demultiplexer 100P like the demultiplexer 100L shown in Fig. 15 also includes an array of FIFO memories 120a, 120b, 120c, and so on which serve as the output buffer memories as a plurality of the output buffer memory. Desired payloads are separated by the separator 106D from the bit stream STM received from the shift register 102 and saved in the corresponding FIFO memories assigned to the destinations determined by the output destination determining unit 105. The video or audio data of the payloads saved in the FIFO memories are then received by one single media processor 122 where they are decoded.

The demultiplexer 100P like the demultiplexer 100N shown in Fig. 17 permits the command memory 111 to be written with the micro-codes received via the host interface 115 from the external CPU. Consequently, whenever the multiplexing format is changed from one to another, its corresponding micro-codes are downloaded

from the external CPU to the command memory 111. As a result, the cost up due to the increase of the memory capacity of the command memory 111 can be avoided and different types of the multiplexing format can equally be handled.

In the embodiments mentioned above, the micro-code read out from the command memory 111 is used for generating a control signal (a command) for actuating one or more components at one time. More specifically, the commands are not sequential but in parallel.

For example, the following is a procedure of processing the DVB packets with the use of sequential commands in the demultiplexer 100N shown in Fig. 17, comprising the steps of:

- (1) registering a received data to the shift register (a shift command);
- (2) examining whether or not the received data is "47" (a comparison command);
- (3) when the step (2) judges "yes", advancing to the step (4) or when "no", returning to the step (1) (a branch command);
- (4) registering another data (a shift command);
- (5) examining "transport_error_indicator" for error (a comparison command);
- (6) when the step (5) judges "yes", discarding the data or when "no", advancing to the step (7) (a branch command);

- (7) downloading "payload_unit_start_indicator" (a load command);
- (8) registering a further data (a shift command);
- (9) downloading a PID data (a load command);
- (10) examining the PID data for registering (a PID comparison command);
- (11) setting the counter with the remaining of a packet length (a load command);
- (12) when the step (10) judges "yes", advancing to the step (13) or when "no", discarding the data (a branch command);
- (13) registering a further data (a shift command);
- (14) subtracting one from the packet length (a subtraction command);
- (15) examining "transport_scrambling_control" for scrambling (a comparison command);
- (16) when the step (15) judges scrambled, discarding the data or when not scrambled, advancing to the step (17) (a branch command);
- (17) downloading "adaptation_field_control" (a load command);
- (18) when the step (17) finds "00", discarding the data or when not find "00", advancing to the step (19) (a branch command);
- (19) down-loading "continuity_counter" (a load command);
- (20) comparing with the preceding "continuity_counter" (a

continuity examination command);

(21) when the step (20) finds no continuity, discarding the data or when finds continuity, advancing to the step (22) (a branch command); and

(22) executing the following steps.

On the contrary, the following is a procedure of processing the DVB packets with the use of parallel commands in the demultiplexer 100N shown in Fig. 17, comprising the steps of:

(1) registering a received data to the shift register (a shift command);

(2) while registering another data, examining whether the preceding data is "47" or not and when "yes", advancing to the step (3) or when "no", repeating the step (2) (a shift command and a comparison/branch command);

(3) examining "transport_error_indicator" for error checking and when so, discarding the data or when not, advancing to the step (4) (a comparison/branch command);

(4) registering a further data, downloading "payload_unit_start_indicator" (a shift command and a load command);

(5) down-loading the PID data (a load command);

(6) while examining the PID data, setting the counter with the remaining of the packet length (a PID comparison command and a

load command);

(7) while registering a further data, subtracting one from the packet length and when the step (6) judges so, advancing to the step (8) or when not, discarding the data (a shift command, a subtraction command, and a branch command);

(8) examining "transport_scrambling_control" and when scrambled, discarding the data or when not, advancing to the step (9) (a branch command);

(9) down-loading "adaptation_field_control" (a load command);

(10) when the step (9) finds "00", discarding the data or when not find "00", advancing to the step (11) (a branch command);

(11) downloading "continuity_counter" (a load command);

(12) comparing with the preceding "continuity_counter" (a continuity examination command);

(13) registering a further data, subtracting one from the packet length, and when the step (12) judges no continuity, discarding the data or when finds continuity, advancing the step (14) (a shift command, a subtraction command, and a branch command); and

(14) executing the following steps.

As the commands are released in parallel, the number of the steps can significantly be reduced. This allows not only the demultiplexing process to be carried out at a higher speed but also the micro-codes to be stored in a less area of the command memory

111. Accordingly, the command memory 111 can be reduced in the storage size and the overall cost of the hardware arrangement will be minimized.

According to the present invention, the command memory is provided for storage of the micro-codes. The micro-codes are read out in a sequence from the command memory and used for controlling the action of each component. As a result, the different types of the multiplexing format can be handled by modifying the micro-codes read out from the command memory, hence contributing to the reduction and the cost down of the circuitry arrangement of the demultiplexer.